Atelier git

Bonjour !

Disclaimer

- On est pas des profs !
  - → Pas parfait
  - → Erreurs
  - → Corrigez nous maintenant ou plus tard

Index

- Introduction (git en bref, histoire)
- Commandes de bases + vue d’un répo
- Les branches et autres commandes++

Introduction

Petites questions

- Différences git / github ? Claire pour tout le monde ?
- Tout le monde à un compte github
  - Sait créer un répo, etc ?

Introduction

- Git ?

  → Un logiciel de control de version decentralisé

  control de version autres termes :

  - VCS (pour Version Control System)
  - SCM (pour Source Code Management)

Logiciel de control de version ?

- Gérer les versions
- Optimiser la mémoire
- Savoir qui change quoi et quand
- Savoir quel version le client a sur son PC

Décentralisé / Centralisé

- Décentralisé (distributed):
  - Une copie du répo localement sur son poste.
- Centralisé (centralized):
  - Un répo central auquel il faut se connecter pour travailler.

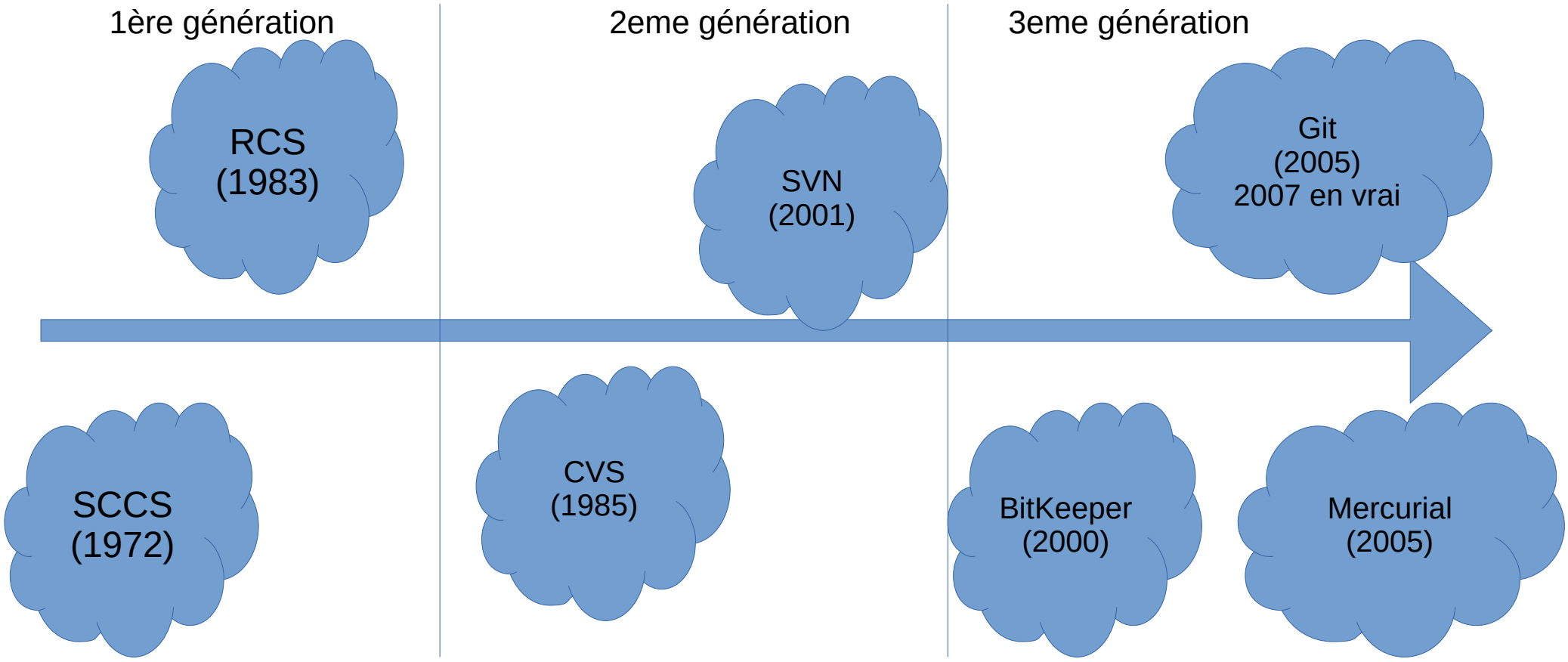
Histoire des VCS/SCM

Les générations

Generation	Networking	Operations
1ère	Aucun	Un fichier à la fois
2ème	Centralisé	Plusieurs fichiers
3ème	Décentralisé	Changeset

Source : https://ericsink.com/vcbe/html/history_of_version_control.html

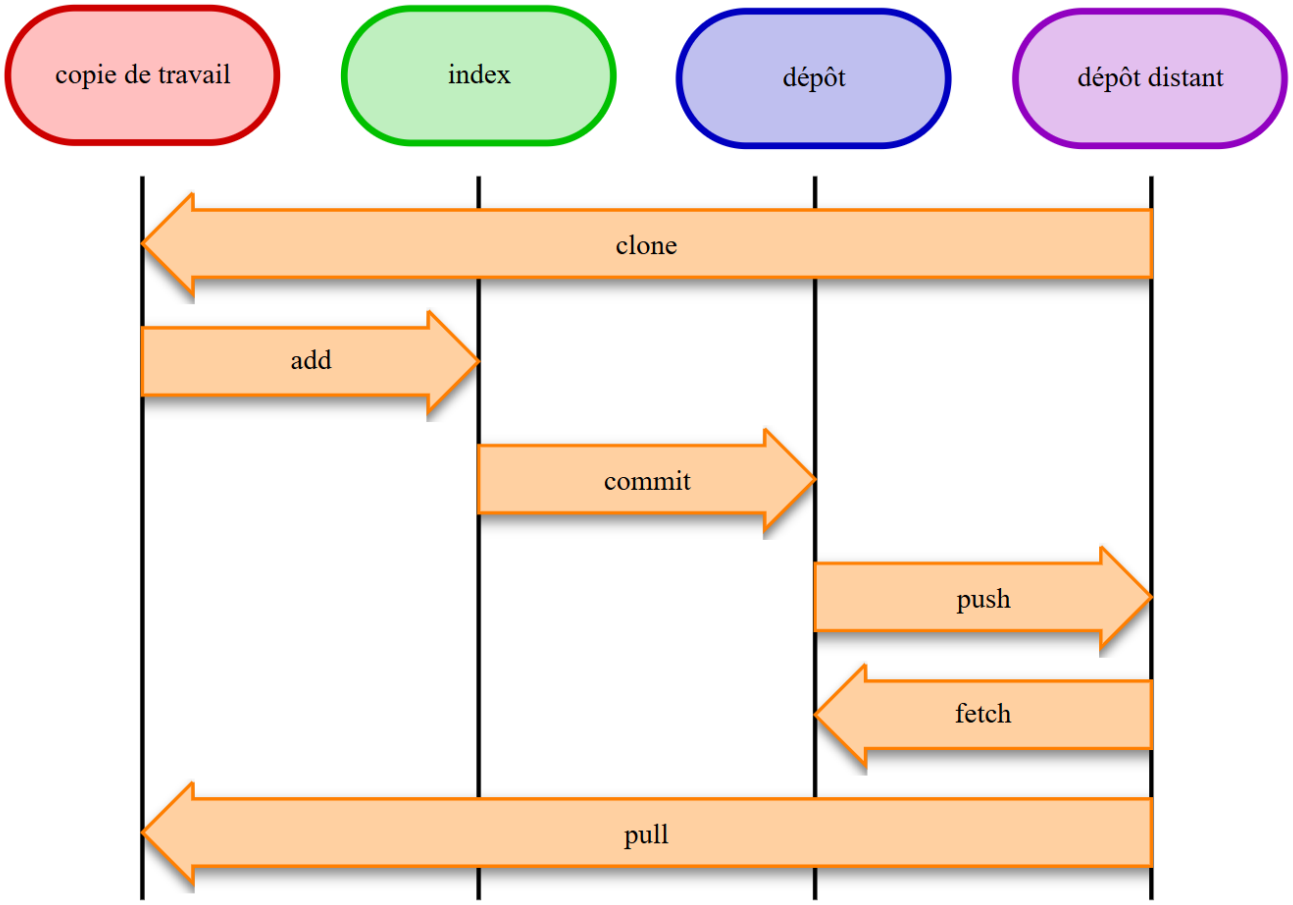
Histoire des VCS/SCM



Histoire des VCS/SCM

Pour plus de détail :
sur le repo github et dans les ressources

Vue d’ensemble d’un répo



Commandes de base

- git clone

```
git clone <remote> [dir]
```

Commandes de base

- git add

```
echo "salut" > msg.txt
git add msg.txt
git commit -m "add: file msg.txt"
git push
rm msg.txt
git add msg.txt
git commit -m ... && git push
```

Commandes de base

- git status

```
On branch jules
Your branch is up to date with 'origin/jules'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   main.c

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   includes/graphics.h
        modified:   includes/mlx_setup.h
        modified:   src/mlx_setup/mlx_setup.c
        modified:   src/parsing/is_normed.c
        modified:   src/parsing/parsing.c

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        wrong_file/091_many_coma.rt
```

Commandes de base

- git commit

```
git commit -m [msg]
```

Commandes de base

Pour ces commandes plus de détails sur le github

- git push
- git fetch
- git pull

Commandes de base

- git remote

```
git remote add one_name link_to_repo
```

```
git remote add another_repo git@vogsphere.42lyon.fr:vogsphere/intra-u
git remote -v
```

```
another_repo    git@vogsphere.42lyon.fr:vogsphere/intra-uuid-9e576b52-bcb5-44a6-9e18-7c73fbdcf19d-7028121-rorollin (fetch)
another_repo    git@vogsphere.42lyon.fr:vogsphere/intra-uuid-9e576b52-bcb5-44a6-9e18-7c73fbdcf19d-7028121-rorollin (push)
```

```
git push another_repo
```

Commandes de base

- git diff

```
git diff msg.txt
echo "salut toi"  > msg.txt
git add msg.txt
git commit -m [msg]
echo "salut en retour"  > msg.txt
git diff
```

```
diff --git a/msg.txt b/msg.txt
index 3d14fea..e347c39 100644
--- a/msg.txt
+++ b/msg.txt
@@ -1 +1 @@
-salut toi
+salut en retour
```

Commandes de base

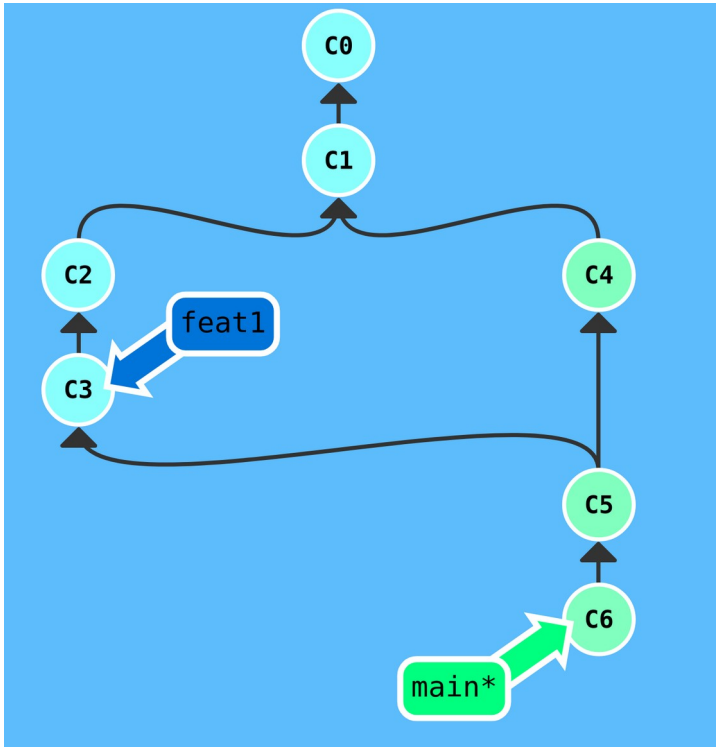
- git log

```
commit 8a35b94acc71063f1654c19968391d2f6fe9635b (HEAD -> main)
Author: jweber <jweber@student.42Lyon.fr>
Date:   Thu Nov 13 13:34:53 2025 +0100

    coucou
```

Les branches

C’est quoi ?

→ une serie de modification parallèle



Les branches

A quoi ça sert ?

- Tester
- Travailler en groupe
- Optimiser son code

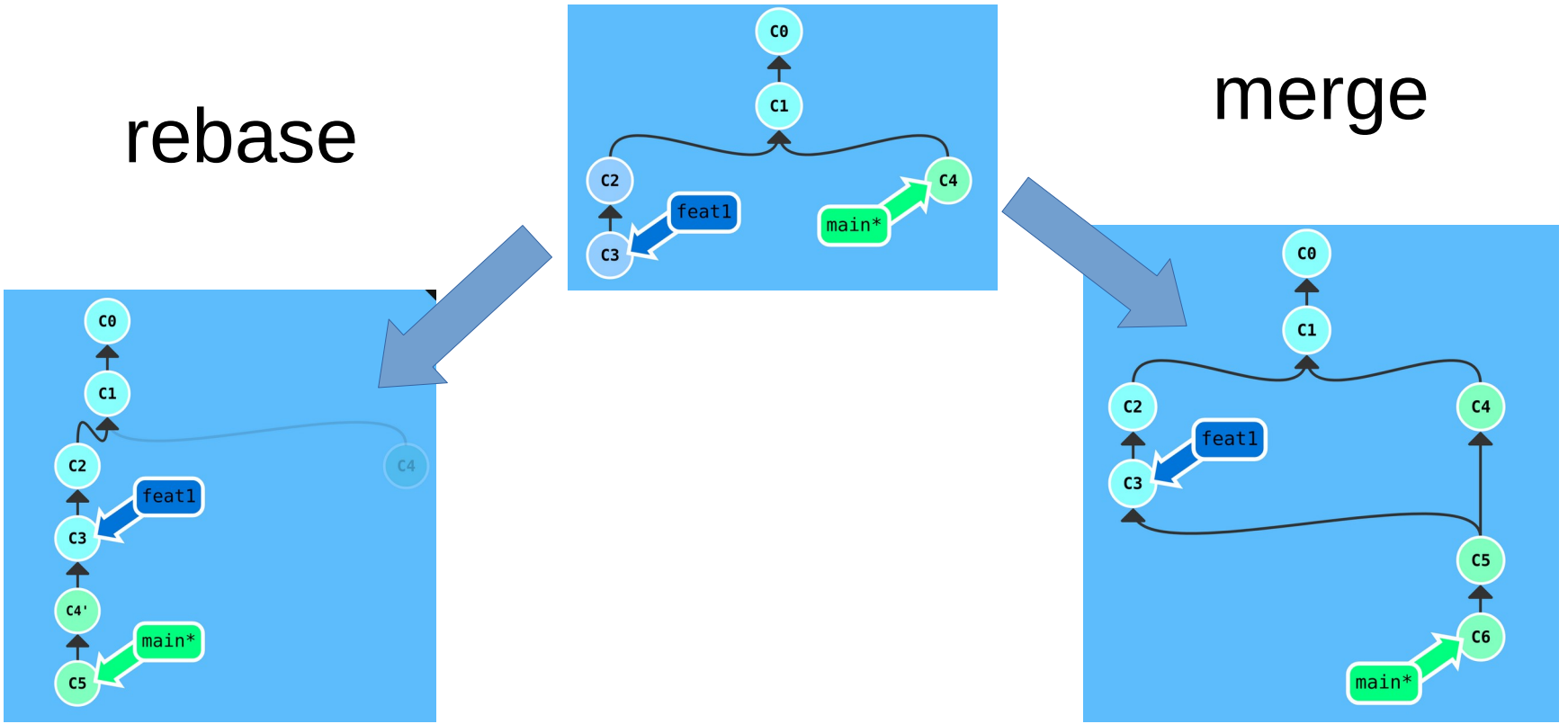
Les branches

Les commandes de bases (branches)

```
# Voir les branches existantes
git branch

# Créer une nouvelle branche
git branch nom-de-branche

# Changer de branche
git switch nom-de-branche

# Créer et passer directement sur une nouvelle branche
git switch -c nom-de-branche

# Fusionner une branche dans la branche actuelle
git merge nom-de-branche

# Supprimer une branche devenue inutile
git branch -d nom-de-branche
```

Les branches

La fusion de branche (merge vs rebase)



Les branches

La fusion de branche (conflits)

Le(s) fichier(s) concernées:

```
CONFLICT (content): Merge conflict in main.c
```

Dans le(s) fichier(s) concernées:

```
<<<<<<< HEAD
printf("Bonjour\n");
=======
printf("Hello\n");
>>>>>>> feature/anglais
```

Récuperer la version d’un fichier

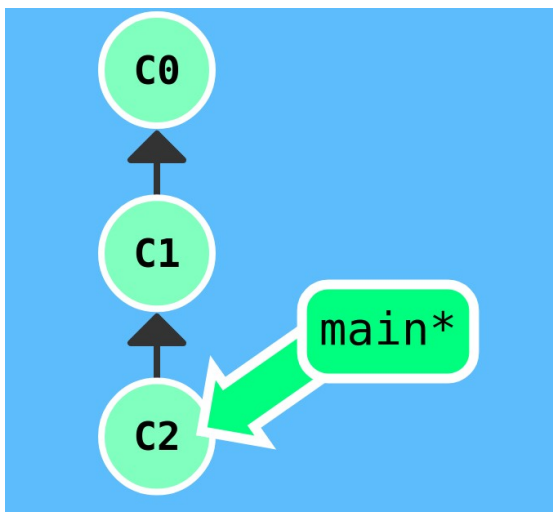
(git checkout)

```
git checkout [hash] [path/to/file]
```

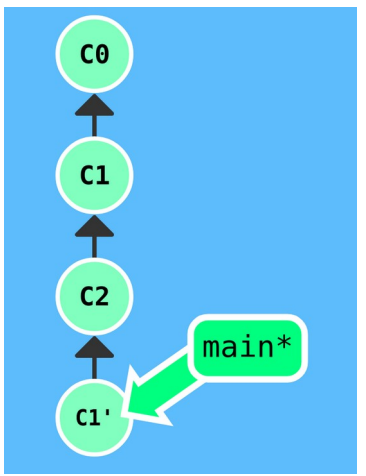
Revenir à un ancien commit

(git revert)

```
git revert <hash_du_commit>
```



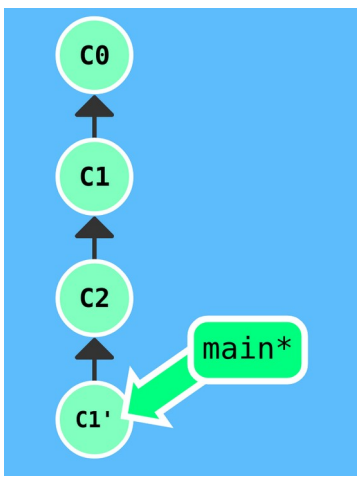
git revert C1



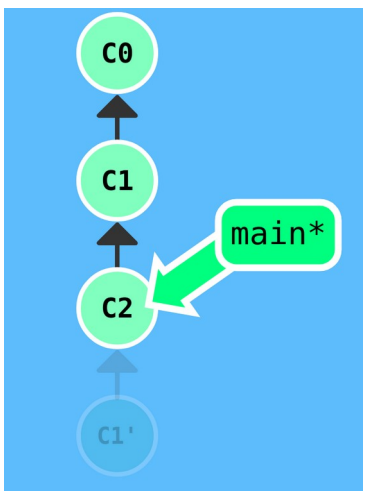
Revenir à un ancien commit
(git reset)

```
# Déplace HEAD, garde les fichiers
git reset --soft <id>

# Supprime aussi les changements du staging
git reset --mixed <id>

# Efface tout (⚠️ dangereux)
git reset --hard <id>
```

Default : --mixed



git reset C2



FIN